

StringCch_W

Carefully specify appropriate units.

Sean Barnum, Digital, Inc. [vita¹]

Copyright © 2007 Digital, Inc.

2007-04-23

Part "Original Digital Coding Rule in XML"

Mime-type: text/xml, size: 7367 bytes

Attack Category	<ul style="list-style-type: none">• Malicious Input
Vulnerability Category	<ul style="list-style-type: none">• Multibyte Character• Buffer Overflow• Unconditional
Software Context	<ul style="list-style-type: none">• String Management• String Formatting
Location	
Description	<p>The StringCch____ string functions all take a *number of characters* parameter, not *number of bytes*. Failure to specify parameters using appropriate units can result in buffer overflows.</p> <p>The StringCch____ and StringCb____ functions are "safer" replacements for standard string manipulation functions. Functions ending in "W" operate on "wide" 2-byte Unicode characters, while functions ending in "A" operate on 1-byte ASCII characters. Functions without either of these suffixes are generic and might operate on wide or narrow characters, depending on compilation settings.</p> <p>Character and byte counts are different when wide characters are involved. This can result in problems when the two types of counts are confused for functions operating on wide characters. This may occur with either "W" functions or generic functions.</p> <p>The "StringCch____W" functions take a character count (e.g., 200 characters = 400 bytes), where the "StringCb____W" functions take a byte count (e.g., 400 bytes for a 200 character buffer).</p> <p>In the case where the user mistakenly passes in a *byte count* instead of a *character count*, the Cch routines will think they have twice the space that is actually there. Assume the user is using a 200 character buffer (which is 400 bytes in size). You want to avoid a situation where the routine thinks it</p>

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)

	<p>has space for 400 wide characters (which would be 800 bytes). Hence, the following call would be bad:</p> <pre>WIDECHAR dest[200]; StringCchCopy(dest, sizeof(dest), src);</pre> <p>In this case sizeof(dest) returns 400 bytes, which is passed into the routine, which thinks it now has 400 *characters* (800 bytes) of space. The buffer will be overrun if src is large, thwarting the bounds checking.</p>																
APIs	<table border="1"> <thead> <tr> <th>Function Name</th><th>Comments</th></tr> </thead> <tbody> <tr> <td>StringCchCatNW</td><td></td></tr> <tr> <td>StringCchCatW</td><td></td></tr> <tr> <td>StringCchCopyNW</td><td></td></tr> <tr> <td>StringCchCopyW</td><td></td></tr> <tr> <td>StringCchGetsW</td><td></td></tr> <tr> <td>StringCchPrintfW</td><td></td></tr> <tr> <td>StringCchVPrintfW</td><td></td></tr> </tbody> </table>	Function Name	Comments	StringCchCatNW		StringCchCatW		StringCchCopyNW		StringCchCopyW		StringCchGetsW		StringCchPrintfW		StringCchVPrintfW	
Function Name	Comments																
StringCchCatNW																	
StringCchCatW																	
StringCchCopyNW																	
StringCchCopyW																	
StringCchGetsW																	
StringCchPrintfW																	
StringCchVPrintfW																	
Method of Attack	By passing long multibyte strings, an attacker could cause a buffer overflow.																
Exception Criteria																	
Solutions	<table border="1"> <thead> <tr> <th>Solution Applicability</th><th>Solution Description</th><th>Solution Efficacy</th></tr> </thead> <tbody> <tr> <td>When any of the StringCch_____ or StringCch_____ functions is called.</td><td> <p>Fix this problem by either using StringCb____W replacement function with the byte size parameter or by carefully ensuring NOT to use simply sizeof(buffer) for size computation. A more general computation is to use sizeof(buffer)/sizeof(buffer[0]) as the "count" computation. This works for locally allocated buffers.</p> </td><td>Effective if done correctly.</td></tr> </tbody> </table>	Solution Applicability	Solution Description	Solution Efficacy	When any of the StringCch_____ or StringCch_____ functions is called.	<p>Fix this problem by either using StringCb____W replacement function with the byte size parameter or by carefully ensuring NOT to use simply sizeof(buffer) for size computation. A more general computation is to use sizeof(buffer)/sizeof(buffer[0]) as the "count" computation. This works for locally allocated buffers.</p>	Effective if done correctly.										
Solution Applicability	Solution Description	Solution Efficacy															
When any of the StringCch_____ or StringCch_____ functions is called.	<p>Fix this problem by either using StringCb____W replacement function with the byte size parameter or by carefully ensuring NOT to use simply sizeof(buffer) for size computation. A more general computation is to use sizeof(buffer)/sizeof(buffer[0]) as the "count" computation. This works for locally allocated buffers.</p>	Effective if done correctly.															

		For buffers allocated on the heap, use a symbolic constant for the number of characters both when allocating the buffer and when manipulating it.
Signature Details	Any of the indicated functions is called.	
Examples of Incorrect Code	<pre>const DWORD BUFBYTES= 14; TCHAR buf1[BUFBYTES/ sizeof(TCHAR)]; LPTSTR buf2 = (LPTSTR)malloc(BUFBYTES); // Local buffer - simple use of sizeof() is wrong for wide characters if (FAILED(StringCchCopy(buf1, sizeof(buf1), TEXT("Words")))) { /* handle error */ } // Buffer on heap - use of buffer size in bytes is wrong for wide characters if (FAILED(StringCchCopy(buf2, BUFBYTES, TEXT("Words")))) { /* handle error */ }</pre>	
Examples of Corrected Code	<pre>const DWORD BUFCHARS = 7; TCHAR buf1[BUFCHARS]; LPTSTR buf2 = (LPTSTR)malloc(BUFCHARS*sizeof(TCHAR)); // Local buffer - can compute buffer size if (FAILED(StringCchCopy(buf1, sizeof(buf1)/sizeof(buf1[0]), TEXT("Words")))) { /* handle error */ } // Buffer on heap - must know buffer size if (FAILED(StringCchCopy(buf2, BUFCHARS, TEXT("Words")))) { /* handle error */ }</pre>	
Source References	<ul style="list-style-type: none"> • http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/resources(strings/stringreference/stringfunctions/stringcchcatn.asp² 	

	<ul style="list-style-type: none"> • http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/resources(strings/stringreference/stringfunctions/stringcchcat.asp³ • http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/resources(strings/stringreference/stringfunctions/stringcchcopyn.asp⁴ • http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/resources(strings/stringreference/stringfunctions/stringcchcopy.asp⁵ • http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/resources(strings/stringreference/stringfunctions/stringcchgets.asp⁶ • http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/resources(strings/stringreference/stringfunctions/stringcchprintf.asp⁷ • http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/resources(strings/stringreference/stringfunctions/stringcchvprintf.asp⁸ 				
Recommended Resource					
Discriminant Set	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 5px;">Operating System</td><td style="padding: 5px;"> <ul style="list-style-type: none"> • Windows </td></tr> <tr> <td style="padding: 5px;">Languages</td><td style="padding: 5px;"> <ul style="list-style-type: none"> • C • C++ </td></tr> </table>	Operating System	<ul style="list-style-type: none"> • Windows 	Languages	<ul style="list-style-type: none"> • C • C++
Operating System	<ul style="list-style-type: none"> • Windows 				
Languages	<ul style="list-style-type: none"> • C • C++ 				

Cigital, Inc. Copyright

Copyright © Digital, Inc. 2005-2007. Digital retains copyrights to this material.

Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works.

For information regarding external or commercial use of copyrighted materials owned by Digital, including information about “Fair Use,” contact Digital at copyright@digital.com¹.

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. <mailto:copyright@digital.com>